task. The next three sections describe these patterns of behavior in some ant species and the respective systems developed inspired by them.

5.2.2. Ant Foraging Behavior

There is a large variety of ant species over the world. However, a number of them present similar foraging behaviors. Field and laboratory experiments with many species of ants have demonstrated an interesting capability of exploiting rich food sources without loosing the capability of exploring the environment, as well as finding the shortest path to a food source in relation to the nest (exploration × exploitation). Some of these observations have led to the development of models of ant behavior and further to the proposal of computational algorithms for the solution of complex problems. This section reviews one such algorithm.

Although most ants live in colonies, and thus present some form of social behavior, for few species of ants there is evidence of the presence of leaders, templates, recipes, or blueprints playing a role in the development of foraging patterns. Instead, the process appears to be based on local competition among information (in the form of varying concentrations of trail *pheromone*), and is used by individual ants to generate observable collective foraging decisions (Camazine et al., 2001).

The process of trail formation in ant colonies can be easily observable with the simple experiment illustrated in Figure 5.1. Place a dish of sugar solution (food source) in the vicinity of an ants' nest - Figure 5.1(a). After some time, *forager ants* will discover the sugar and shortly after, through a *recruitment* process, a number of foragers will appear at the food source - Figure 5.1(b). Observation will reveal ants trafficking between the nest and the food source as if they were following a *trail* on the ground - Figure 5.1(c). Note the presence of a few foragers not following the trail; an important behavior for the location of alternative food sources; that is, exploration of the environment.

Generally speaking, *recruitment* is a behavioral mechanism that enables an ant colony to assemble rapidly a large number of foragers at a desirable food source and to perform efficient decision making, such as the selection of the most profitable food source or the choice of the shortest path between the nest and the food source. Different recruitment mechanisms exist (Deneubourg et al., 1986). In *mass recruitment*, a scout (*recruiter*) discovers the food source and returns to the nest, laying a chemical (pheromone) trail. In the nest, some of its nestmates (the recruited) detect the trail and follow it to the food source. There they ingest food and return to the nest *reinforcing* the trail. In *tandem recruitment*, the scout invites ants at the nest to accompany her back to the food. One recruit succeeds in following the leader, the two ants being in close contact. In *group recruitment*, the recruiter leads a group of recruits to the food source by means of a short-range chemical attractant. After food ingestion, in each recruitment type, the recruited become recruiters. This typical self-reinforcing (positive feedback) process is the basis of many activities of ants' societies. The recruitment process slows down when there are fewer ants left to be recruited or when there are other forces, such as alternative food sources, competing for the ants' attention.

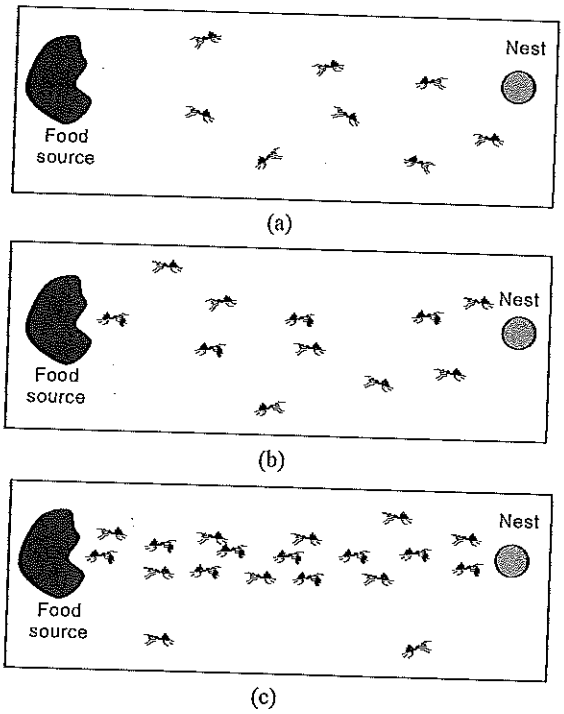

Figure 5.1: In the collective foraging of some ant species, ants recruit nestmates by releasing pheromone on the path from the food source to the nest; a pheromone trail is thus established. (a) Foraging ants. (b) A few ants find the food source and start recruiting nestmates by releasing pheromone. (c) A pheromone trail is formed.

The process of trail formation may be divided in two steps: *trail-laying* and *trail-following* (Camazine et al., 2001). Consider the particular case of mass recruitment and some of the experimental set ups used to observe it and propose mathematical models. The behavior of individual ants during trail recruitment starts when a forager ant, on discovering a food source, returns to the nest and lays a chemical trail all the way home. These chemicals are low molecular weight substances, called *pheromone*, produced in special glands or in the guts. Other ants, either foraging or waiting in the nest, are then stimulated under the influence of the pheromone to start exploiting the food source encountered. In addition to the pheromone trail, recruiter ants may provide other stimuli. The ants recruited follow the trail to the food source and load up with food. On their return journey to the nest, they add their own pheromone to the trail and may even provide further stimulation to other foragers in the nest and on the way. The pheromone thus has two important functions: 1) primarily to define the trail, and also 2) to serve as an orientation signal for ants traveling outside the nest. It is important to remark, however, that the pheromone evaporates, usually very slowly, with time. It means that, for example, if the food source is completely

depleted, the trail will disappear after some time due to the lack of reinforcement.

Goss et al. (1989) and Deneubourg et al. (1990) performed a number of experiments and demonstrated that the trail-laying trail-following behavior of some ant species enables them to find the shortest path between a food source and the nest. They used a simple yet elegant experimental set up with the Argentine ant *Iridomyrmex humilis*. Basically, their experiment can be summarized as follows. Laboratory colonies of *I. humilis* were given access to a food source in an arena linked to the nest by a bridge consisting of two branches of different lengths, arranged so that a forager going in either direction (from the nest to the food source or vice-versa) must choose between one or the other branch, as illustrated in Figure 5.2.

One experimental observation is that, after a transitory phase that can last a few minutes, most of the ants choose the shortest path. It is also observed that an ant's probability of selecting the shortest path increases with the difference in length between the two branches. The explanation for this capability of selecting the shortest route from the nest to the food source comes from the use of pheromone as an indirect form of communication, known as *stigmergy*, mediated by local modifications of the environment. Before starting a discussion about stigmergy in the process of pheromone trail laying and following, it must be asked what would happen if the shorter branch were presented to the colony after the longer one. In this case, the shorter path would not be selected because the longer branch is already marked with a pheromone trail. However, if the pheromone evaporates quickly enough, longer paths would have trouble to maintain stable pheromone trails. This is not the case for most ant species, but it is the approach usually taken by engineers and computer scientists in their computational implementations of *ant colony optimization* (ACO) algorithms.

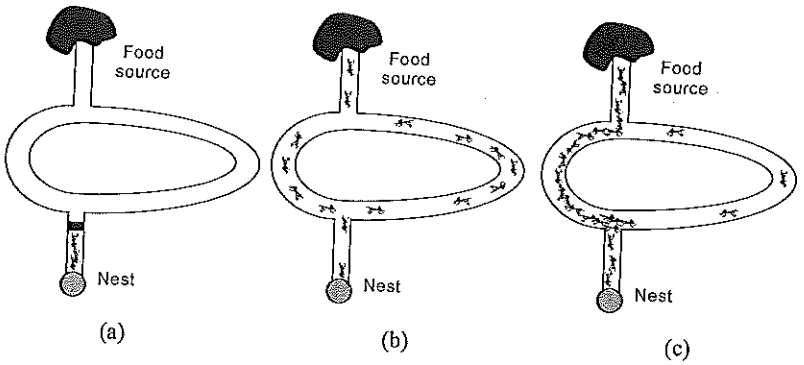

Figure 5.2: An experimental set up that can be used to demonstrate that the ant *I. Humilis* is capable of finding the shortest path between the nest and a food source. (b) The bridge is initially closed. (b) Initial distribution of ants after the bridge is open. (b) Distribution of ants after some time has passed since they were allowed to exploit the food source.

Another observation is that randomness plays an important role in ant foraging. Ants do not follow pheromone trails perfectly. Instead, they have a probabilistic chance of loosing their way as they follow trails. Deneubourg et al. (1986) argue that this 'ant randomness' is not a defective stage on an evolutionary path 'towards an idealistic deterministic system of communication.' Rather, this randomness is an evolutionarily adaptive behavior. In some of their laboratory experiments, they describe one case with two food sources near an ant nest: a rich food source far from the nest, and an inferior source close to the nest. Initially, the ants discover the inferior food source and form a robust trail to that source. But some ants wander off the trail. These lost ants discover the richer source and form a trail to it. Since an ant's pheromone emission is related to the richness of the food source, the trail to the richer source becomes stronger than the original trail. Eventually, most ants shift to the richer source. Therefore, the randomness of the ants provides a way for the colony to explore multiple food sources in parallel; that is, randomness allows for exploration. While positive feedback through pheromone trails encourages exploitation of particular sources, randomness encourages exploration of multiple sources.

Stigmergy

The case of foraging for food in the Argentine ants shows how stigmergy can be used to coordinate the ant's foraging activities by means of self-organization. Self-organized trail-laying by individual ants is a way of modifying the environment to communicate to other nestmates to follow that trail. In the experimental set up of Figure 5.2, when the ants arrive at the branching point, they have to make a probabilistic choice between one of the two branches to follow. Such choice is biased by the amount of pheromone they smell on the branches. This behavior has a self-reinforcing (positive feedback) effect, because choosing a branch will automatically increase the probability that this branch will be chosen again by other ants. At the beginning of the experiment there is no pheromone on the two branches and thus, the choice of a branch will be made with the same probability for both branches. Due to the different branch lengths, the ants traveling on the shortest branch will be the first ones to reach the food source. In their journey back to the nest, these ants will smell some pheromone trail (released by themselves) on the shorter path and will thus follow it. More pheromone will then be released on the shorter path, making it even more attractive for the other ants. Therefore, ants more and more frequently select the shorter path.

5.2.3. Ant Colony Optimization (ACO)

The choice of the shortest path enables ants to minimize the time spent traveling between nest and food source, thus taking less time to complete the route. This also allows ants to collect food more quickly and minimize the risk that this food source is found and monopolized by a stronger competitor, such as a larger colony. Shorter paths also mean lower transportation costs (Camazine et al., 2001).

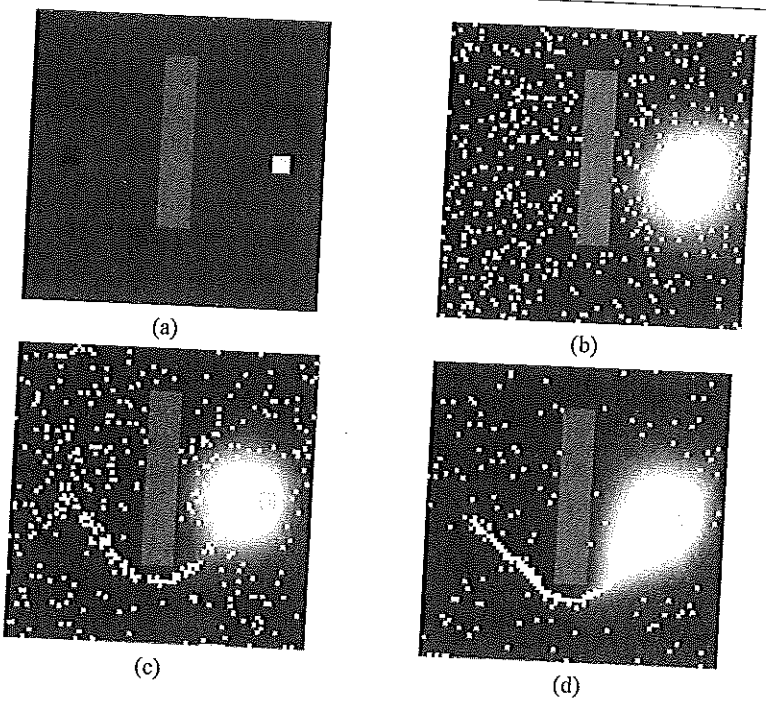

Figure 5.3: Artificial life simulation of pheromone trail laying and following by ants. (a) Environmental setup. The square on the left corresponds to the ants' nest, and the one on the right is the food source. In between the two there is an obstacle whose top part is slightly longer than the bottom part. (b) 500 ants leave the nest in search for food, and release pheromone (depicted in white color) while carrying food back to the nest. (c) The deposition of pheromone on the environment serves as a reinforcement signal to recruit other ants to gather food. (d) A strong pheromone trail in the shorter route is established.

To illustrate this, consider the artificial life simulation (Chapter 8) of pheromone trail-laying, trail-following illustrated in Figure 5.3. In this simulation, a food source is available (big square on the right hand side of the picture) together with an ant nest (big square on the left hand side of the picture) and an obstacle between them (rectangle centered). At first, the virtual ants explore the environment randomly. After they find the food source and start carrying it back to the nest, those ants that choose the shorter path to the nest return first, thus reinforcing the shorter path. Note that the top part of the obstacle is slightly longer than its bottom part, indicating that the shorter route from the nest to the food source is the bottom route. Next, the ants maintain almost only the shortest trail from the food source to the nest, leading to the exploitation of this trail.

The problem of finding the shortest route from the nest to the food source is akin to the well-known traveling salesman problem (TSP), discussed in Chapter 3. The salesman has to find the shortest route by which to visit a given number of cities, each exactly once. The goal is to minimize the cost (distance) of travel.

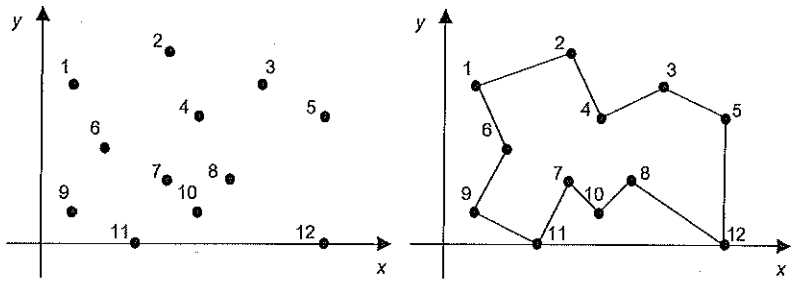

Figure 5.4: A simple instance of the TSP. (a) A set with 12 cities. (b) A minimal route connecting the cities.

Inspired by the experiments of Goss et al. (1989, 1990), Dorigo and collaborators (Dorigo et al., 1996) extended this ant model to solve the traveling salesman problem. Their approach relies on *artificial ants* laying and following *artificial pheromone trails*. Assume the simple TSP instance of Figure 5.4(a). A colony of artificial ants, each independently going from one city to another, favoring nearby cities but otherwise traveling randomly. While traversing a link, i.e., a path from one city to another, an ant deposits some pheromone on it. The amount of pheromone being inversely proportional to the overall length of the tour: the shorter the tour length, the more pheromone released; and vice-versa. After all the artificial ants have completed their tours and released pheromone, the links belonging to the highest number of shorter tours will have more pheromone deposited. Because the pheromone evaporates with time, links in longer routes will eventually contain much less pheromone than links in shorter tours. As the colony of artificial ants is allowed to travel through the cities a number of times, those tours less reinforced (by pheromone) will attract fewer ants in their next travel (Bonabeau and Théraulaz, 2000). Dorigo and collaborators (Dorigo et al., 1996) have found that by repeating this process a number of times, the artificial ants are able to determine progressively shorter routes, such as the one illustrated in Figure 5.4(b).

The Simple Ant Colony Optimization Algorithm (S-ACO)

Ant algorithms were first proposed by Dorigo et al. (1991) and Dorigo (1992) as a multi-agent approach to solve discrete optimization problems, such as the traveling salesman problem and the quadratic assignment problem (QAP). Similarly to all the other fields of investigation discussed in this book, there is a lot of ongoing research. Therefore, there are various versions and extensions of ant algorithms. This section reviews the simplest ACO algorithm, and the next section presents a general-purpose ACO algorithm along with its most relevant features.

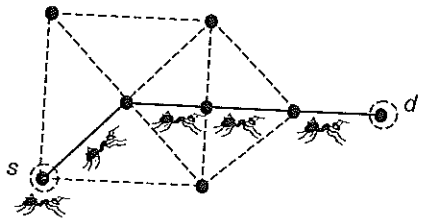

Figure 5.5: An ant travels through the graph from the source node s to the destination node d thus building a solution (path). (Note that this travel resembles the one performed by the traveling salesperson, who has a number of cities to visit and has to decide which path to take.)

Assuming a connected graph $G = (V, E)$, the simple ACO algorithm (S-ACO) can be used to find a solution to the shortest path problem defined on the graph G. (Appendix B.4.4 brings the necessary fundamentals from *graph theory*.) A solution is a path on the graph connecting a source node s to a destination node d and the path length is given by the number of edges traversed (Dorigo and Di Caro, 1999), as illustrated in Figure 5.5. Associated with each edge (i,j) of the graph there is a variable τ_{ij} termed *artificial pheromone trail*, or simply pheromone. Every *artificial ant* is capable of "marking" an edge with pheromone and "smelling" (reading) the pheromone on the trail.

Each ant traverses one node per iteration step t and, at each node, the local information about its pheromone level, τ_{ij}, is used by the ant such that it can probabilistically decide the next node to move to, according to the following rule:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}(t)}{\sum_{j \in N_i} \tau_{ij}(t)} & \text{if } j \in N_i \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

where $p_{ij}^{k}(t)$ is the probability that ant k located at node i moves to node j, $\tau_{ij}(t)$ is the pheromone level of edge (i,j), all taken at iteration t, and N_i is the set of one step neighbors of node i.

While traversing an edge (i,j), the ant deposits some pheromone on it, and the pheromone level of edge (i,j) is updated according to the following rule:

$$\tau_{ij}(t) \leftarrow \tau_{ij}(t) + \Delta\tau \tag{5.2}$$

where t is the iteration counter and $\Delta\tau$ is the constant amount of pheromone deposited by the ant.

The analysis of Equation (5.1) and Equation (5.2) lead to the conclusion that when an ant deposits some pheromone on a given edge (i,j), it increases the probability that this edge is selected by another ant, thus reinforcing the trail that passes through this edge. The presence of positive feedback favoring the selection of short paths is clear in this simple model.

Preliminary experiments with the S-ACO algorithm demonstrated that it is capable of determining the shortest route between a simulated nest and food source in a computer simulation similar to the laboratory experiment illustrated in Figure 5.2. However, the behavior of the algorithm becomes unstable when the complexity of the searched graph increases (Dorigo and Di Caro, 1999). In order to overcome this limitation and to avoid a quick convergence of all ants towards sub-optimal routes, an evaporation of the pheromone trails was allowed, thus changing Equation (5.2) into:

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau \qquad (5.3)$$

where $\rho \in (0,1]$ is the pheromone decay rate.

General-Purpose Ant Colony Optimization Algorithm

Ant algorithms constitute all algorithms for discrete optimization that took inspiration from the observation of the foraging behavior of ant colonies (Dorigo et al., 1999). Although some authors have already developed versions of ant algorithms for continuous optimization (e.g., Bilchev and Parmee, 1995), not much research has been conducted in this direction and, thus, the focus of the discussion to be presented here is on ACO for discrete optimization.

An ACO algorithm alternates, for a maximum number of iterations `max_it`, the application of two basic procedures (Bonabeau et al., 1999). Assume a search is being performed on a connected graph $G = (V,E)$ with V nodes and E edges:

1. A parallel solution construction/modification procedure in which a set of N ants builds/modifies in parallel N solutions to the problem under study.
2. A pheromone trail updating procedure by which the amount of pheromone trail on the problem graph edges is changed.

The process of building or modifying a solution is made in a probabilistic fashion, and the probability of a new edge to be added to the solution being built is a function of the edge *heuristic desirability* η, and of the *pheromone trail* τ deposited by previous ants. The heuristic desirability η expresses the likelihood of an ant moving to a given edge. For instance, in cases the minimal path is being sought among a number of edges, the desirability η is usually chosen to be the inverse of the distance between a pair of nodes. The modification (updating) of pheromone trails is a function of both the evaporation rate ρ (see Equation (5.3)) and the quality of the solutions produced. Algorithm 5.1 depicts the standard, or general-purpose, ACO algorithm to perform discrete optimization (Bonabeau et al., 1999). The parameter e is the number of edges on the graph, and `best` is the best solution (path) found so far; the pheromone level of each edge is only updated after all ants have given their contributions (laid some pheromone). The generic algorithm assumes that the heuristic desirability function of an edge and how to update the pheromone trail have already been defined.

```
procedure [best] = ACO(max_it, N, τ0)
    initialize τij   //usually initialized with the same τ0
    initialize best
    place each ant k on a randomly selected edge
    t ← 1
    while t < max_it do,
       for i = 1 to N do,  //for each ant
           build a solution by applying a probabilistic
           transition rule (e-1) times.
           //The rule is a function of τ and η
           //e is the number of edges on the graph G
       end for
       eval the cost of every solution built
       if an improved solution is found,
          then update the best solution found
       end if
       update pheromone trails
       t ← t + 1
   end while
end procedure
```

Algorithm 5.1: Standard ACO for discrete optimization.

Selected Applications from the Literature: A Brief Description

Algorithm 5.1 presents the general-purpose ACO algorithm. It is important to note the basic features of an ACO algorithm:

- A colony of ants that will be used to build a solution in the graph.
- A probabilistic transition rule responsible for determining the next edge of the graph to which an ant will move.
- A heuristic desirability that will influence the probability of an ant moving to a given edge.
- The pheromone level of each edge, which indicates how 'good' it is.

To illustrate how to use this simple algorithm in practical problems, consider the following two examples. The first example - traveling salesman problem - was chosen because it was one of the first tasks used to evaluate an ACO algorithm. In addition, the TSP is a shortest path problem to which the ant colony metaphor is easily adapted, it is an NP-hard problem, it is didactic, and it has been broadly studied (Lawer et al., 1985). The second example - vehicle routing - illustrates how a small variation in the ACO for the traveling salesman problem allows it to be applied to the vehicle routing problem.

Traveling Salesman

In the TSP the goal is to find a closed tour of minimal length connecting e given cities, where each city must be visited once and only once. Let d_{ij} be the Euclidean distance between cities i and j given by:

$$d_{ij} = [(x_i - x_j)^2 + (y_i - y_j)^2]^{1/2} \tag{5.4}$$

where x_m and y_m are the coordinates of city m, on the x-y plane. This problem can be more generally defined on a graph $G = (V,E)$, where the cities correspond to the nodes V and the connections between them are the edges E.

In (Dorigo et al., 1996; Dorigo and Gambardella, 1997a,b), the authors introduced the Ant System (AS) as an ACO algorithm to solve the TSP problem. In AS ants build solutions to the TSP by moving on the problem graph from one city to another until they complete a tour. During each iteration of the AS, an ant k, $k = 1, \ldots, N$ (a colony of N ants is assumed), builds a tour by applying a probabilistic transition rule $(e - 1)$ times, as described in Algorithm 5.1. The iterations are indexed by the iteration counter t and a maximum number of iteration steps, `max_it`, is defined for the iterative procedure of adaptation, i.e., $1 \le t \le$ `max_it`. The description to be presented here follows those of (Dorigo et al., 1996; Dorigo and Gambardella, 1997a,b; Bonabeau et al., 1999).

For each ant, the transition from city i to city j at iteration t depends on: 1) the fact that the city has already been visited or not; 2) the inverse of the distance d_{ij} between cities i and j, termed *visibility* $\eta_{ij} = 1/d_{ij}$; and 3) the amount of pheromone τ_{ij} on the edge connecting cities i and j. As in the TSP problem each ant has to visit a city only once, some knowledge of the edges (cities) already visited has to be kept by the ants, thus they must possess some sort of memory, also called a *tabu list*. The memory is used to define the set J_i^k of cities that the ant k still has to visit while in city i.

The probability of an ant k going from city i to city j at iteration t is given by the following transition rule:

$$p_{ij}^k(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{il \in J_i^k} [\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}} & \text{if } j \in J_i^k \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

where $\tau_{ij}(t)$ is the pheromone level of edge (i,j), η_{ij} is the visibility of city j when in city i, and J_i^k is the tabu list of cities still to be visited by ant k from node i. The parameters α and β are user-defined and control the relative weight of trail intensity $\tau_{ij}(t)$ and visibility η_{ij}. For instance, if $\alpha = 0$, the closest cities are more likely to be chosen, and if $\beta = 0$, by contrast, only pheromone amplification is considered.

Similarly to the simple ACO algorithm (S-ACO), while traversing an edge (city), an ant lays some pheromone on that edge. In the AS, the quantity $\Delta\tau_{ij}^k(t)$ of pheromone released on each edge (i,j) by ant k depends on how well it has performed according to the following rule:

$$\Delta\tau_{ij}^k(t) = \begin{cases} Q / L^k(t) & \text{if } (i,j) \in T^k(t) \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

where $L^k(t)$ is the length of the tour $T^k(t)$ performed by ant k at iteration t, and Q is another user-defined parameter.

The pheromone updating rule is the same as the one used for the simple ACO algorithm (Equation (5.3)) taking into account the differential amount of pheromone released by each ant in each iteration:

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{5.7}$$

where $\rho \in (0,1]$ is the pheromone decay rate, $\Delta\tau_{ij}(t) = \sum_k \Delta\tau^k_{ij}(t)$, and $k = 1, \ldots, N$ is the index of ants.

According to Dorigo et al. (1996), the number N of ants is an important parameter of the algorithm. Too many ants reinforce sub-optimal trails leading to bad solutions, whereas too few ants would not result in a sufficient cooperative behavior due to pheromone decay. They suggested to use $N = e$, that is, a number of ants equals to the number of cities.

```
procedure [best] = AS-TSP(max_it, α, β, ρ, N, e, Q, τ0, b)
    initialize τij   //usually initialized with the same τ0
    place each ant k on a randomly selected city
    Let best be the best tour found from the beginning and
    Lbest its length
    t ← 1
    while t < max_it do,
        for i = 1 to N do,          //for every ant
            //e is the number of cities on the graph
            build tour T^k(t) by applying (e-1) times the fol-
            lowing step:
            At city i, choose the next city j with probabil-
            ity given by Equation (5.5)
        end for
        eval the length of the tour performed by each ant
        if a shorter tour is found,
            then update best and Lbest
        end if
        for every city e do,
            Update pheromone trails by applying the rule:
            τij(t+1) ← (1-ρ)τij(t) + Δτij(t) + b.Δτ^b_ij(t), where
            Δτij(t) = Σk Δτ^k_ij(t), k = 1,...N;
            Δτ^k_ij(t) = { Q/L^k(t)   if (i,j) ∈ T^k(t), and
                         { 0          otherwise
            Δτ^b_ij(t) = { Q/Lbest    if (i,j) ∈ best.
                         { 0          otherwise
        end for
        t ← t + 1
    end while
end procedure
```

Algorithm 5.2: Ant system for the traveling salesman problem (AS-TSP).

In an effort to improve the AS performance when applied to the TSP problem, the authors also introduced the idea of "elitist ants", a term borrowed from evolutionary algorithms. An elitist ant is the one that reinforces the edges belonging to the best route found so far, `best`. Such ants would reinforce the trail (cities or edges) of the best tour by adding $b.Q/L_{best}$ to the pheromone level of these edges, where b is the number of elitist ants chosen, and L_{best} is the length of the best tour found from the beginning of the trial. Algorithm 5.2 depicts the ant system for the traveling salesman problem (AS-TSP). The authors employed the following values for the parameters in their experiments: $\alpha = 1$, $\beta = 5$, $\rho = 0.5$, $N = e$, $Q = 100$, $\tau_0 = 10^{-6}$, and $b = 5$.

It is important to remark that the results presented in Dorigo et al. (1996) were interesting but disappointing. The AS-TSP could outperform other approaches, such as a GA, when applied to small instances of the TSP, but its performance was poor when applied to TSP problems with a large number of cities. In order to enhance even further the performance of the ACO approach, the authors altered the transition rule (Equation (5.5)), employed local updating rules of the pheromone trail (Equation (5.6)), and used a candidate list to restrict the choice of the next city to visit (Dorigo and Gambardella, 1997a,b). These modifications made the algorithm competitive in relation to other approaches from the literature. Further improvements, variations, and applications of ant colony optimization algorithms can be found in Corne et al. (1999), Dorigo et al. (2002), Dorigo and Stützle (2004), and de Castro and Von Zuben (2004).

Vehicle Routing

The simplest *vehicle routing problem* (VRP) can be described as follows: given a fleet of vehicles with uniform capacity, a common depot, and several costumer demands (represented as a collection of geographical scattered points), find the set of routes with overall minimum route cost which service all the demands. All the itineraries start and end at the depot and they must be designed in such a way that each costumer is served only once and just by one vehicle. Note that the VRP is closely related to the TSP as presented here; the VRP consists of the solution of many TSPs considering a single common start and end city for each TSP, which now has a limited capacity.

Bullnheimer et al. (1999a,b) and Bullnheimer (1999) modified the AS-TSP algorithm to solve the vehicle routing problem with one central depot and identical vehicles. The VRP problem was represented by a complete *weighted directed graph* $G = (V,E,w)$, where $V = \{v_0, v_1, \ldots, v_N\}$ is the set of nodes of G, $E = \{(v_i,v_j) : i \neq j\}$ its set of edges, and $w_{ij} \geq 0$ is a weight connecting nodes i and j. The node v_0 denotes the depot and the other vertices are the customers or cities. The weights w_{ij}, associated with edge (v_i,v_j), represent the distance (time or cost) between v_i and v_j. A demand $d_i \geq 0$ and a service time $\delta_i \geq 0$ are associated with each customer v_i. The respective values for the depot are $v_0 = d_0 = 0$.

The objective is to find the minimum cost vehicle routes where:

- Every customer is visited only once by only one vehicle.

- All vehicle routes begin and end at the depot.
- For every vehicle route the total demand does not exceed the vehicle capacity D.
- For every vehicle route the total route length, including service times, does not exceed a given threshold L.

Similarly to the AS-TSP, to solve the vehicle routing problem one ant is placed on each node of the graph and the ants construct solutions by sequentially visiting a city, until all cities have been visited. Whenever the choice of another city would lead to an unfeasible solution; that is, vehicle capacity greater than D or total route length greater than the threshold L, the depot is chosen and a new tour is started.

The VRP problem was solved by applying the procedure presented in Algorithm 5.1 with the probabilistic transition rule presented in Equation (5.5), which takes into account the pheromone trail τ_{ij} and the heuristic desirability η_{ij}. The heuristic desirability of a city follows a *parametrical saving function* (Paessens, 1988):

$$\eta_{ij} = w_{i0} + w_{0j} - g.w_{ij} + f.\ |w_{i0} - w_{0j}| \tag{5.8}$$

where g and f are two user-defined parameters.

In the standard ant system algorithm, after an ant has constructed a solution, the pheromone trails are laid by all ants depending on the objective value of the solution. It was discussed in the previous example that by using σ elitist ants, the performance of the algorithm was improved for the TSP problem. In (Bullnheimer et al., 1999b), the authors suggested that by ranking the ants according to the solution quality and using only the best ranked ants, also called elitist ants, to update the pheromone trails would improve even further the performance of the algorithm for the VRP problem. The new updating rule is as follows:

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau^{r}_{ij}(t) + \sigma.\Delta\tau_{ij}^{+}(t) \tag{5.9}$$

where $\rho \in (0,1]$ is the pheromone decay rate. Only if an edge (v_i,v_j) was used by the μ-th best ant, the pheromone trail is increased by a quantity

$$\Delta\tau^{r}_{ij}(t) = \sum_{\mu=1}^{\sigma-1} \Delta\tau^{\mu}_{ij}(t) \tag{5.10}$$

where $\Delta\tau^{\mu}_{ij}(t) = (\sigma - \mu)/L_{\mu}(t)$ if ant μ uses the edge (v_i,v_j), and $\Delta\tau^{\mu}_{ij}(t) = 0$ otherwise. $L_{\mu}(t)$ is the length of the tour ant μ performs at iteration t. All edges belonging to the best tour determined so far are emphasized as if σ elitist ants had used them. Therefore, each elitist ant increases the trail intensity by an amount $\Delta\tau_{ij}^{+}(t)$ that is equal to $1/L^{+}$ if edge (v_i,v_j) belongs to the so far best solution, and zero otherwise.

Scope of ACO Algorithms

Although the traveling salesman problem is an intuitive application of ACO, many other discrete (combinatorial) optimization problems can be solved with

ACO. Bonabeau et al. (2000) claim that ACO is currently the best available heuristic for the sequential ordering problem, for real-world instances of the quadratic assignment problem, and is among the best alternatives for the vehicle and network routing problems. Good surveys of applications of ACO algorithms, including a number of main references can be found in Bonabeau et al. (1999, 2000), Dorigo et al. (1999), Dorigo and Di Caro (1999), and Dorigo and Stützle (2004). The main problems tackled by ACO algorithms are: TSP, network routing, graph coloring, shortest common super sequence, quadratic assignment, machine scheduling, vehicle routing, multiple knapsack, frequency assignment, and sequential ordering.

ACO algorithms are suitable for solving problems that involve graph searching, mainly minimal cost problems. Similarly to all the other approaches discussed in this text, ACO algorithms may be applied only when more classical approaches, such as dynamic programming or other methods cannot be (efficiently) applied. Dorigo and Di Caro (1999) suggest a list of particular cases of interest for ACO algorithms:

- *NP problems*: a problem is said to be solvable in polynomial time if there is an algorithm to solve it in a time that is a polynomial function of the size of the input. NP stands for *nondeterministic polynomial*. It is a large class of problems that have the property that if any solution exists, there is at least one solution which may be verified as correct in polynomial time. Less rigorously, a problem is NP if we can check a proposed solution in polynomial time. A more detailed description of problem complexity is provided in Appendix B.3.3.
- *Static and dynamic combinatorial optimization problems*: a combinatorial problem is the one in which there is a large discrete set of possible solutions. The static problems are those whose characteristics do not change over time, e.g., the classic traveling salesman problem. The shortest path problems in which the properties of the graph representation of the problem change over time constitute the dynamic problems (e.g., network routing).
- *Distributed problems*: those in which the computational architecture is spatially distributed (e.g., parallel or network processing), and may require a set of agents to find suitable solutions. The inherent distribution of agents' resources, such as knowledge and capability, promotes the need of a cooperative work.

From Natural to Artificial Ants

Most of the ideas of ACO were inspired by the foraging behavior of trail-laying trail-following ants. In particular, the main features of ACO algorithms are the use of: 1) a colony of cooperating individuals, 2) a pheromone trail for local indirect (stigmergic) communication, 3) a sequence of local moves to find the shortest paths, and 4) a probabilistic decision rule using local information (Dorigo et al., 1999). Table 5.1 clarifies how the ACO approach uses the biological terminology borrowed from the natural world of ants.

Table 5.1: Interpretation of the biological terminology into the computational Ant Colony Optimization (ACO) algorithms.

Biology (Ant Foraging)	ACO Algorithms
Ant	Individual (agent) used to build (construct) a solution to the problem
Ant colony	Population (colony) of cooperating individuals known as artificial ants
Pheromone trail	Modification of the environment caused by the artificial ants in order to provide an indirect mean of communication with other ants of the colony This allows the assessment of the quality of a given edge on a graph
Pheromone evaporation	Reduction in the pheromone level of a given path (edge) due to aging

5.2.4. Clustering of Dead Bodies and Larval Sorting in Ant Colonies

Chrétien (1996) has performed a number of experiments to study the organization of cemeteries of the ant *Lasius niger*. It has been observed, in several species of ants, workers grouping corpses of ants, or parts of dead ants, to clean up their nests, thus forming cemeteries. Further research on different ant species (Deneubourg et al., 1991) has confirmed the cemetery organization capability of ants.

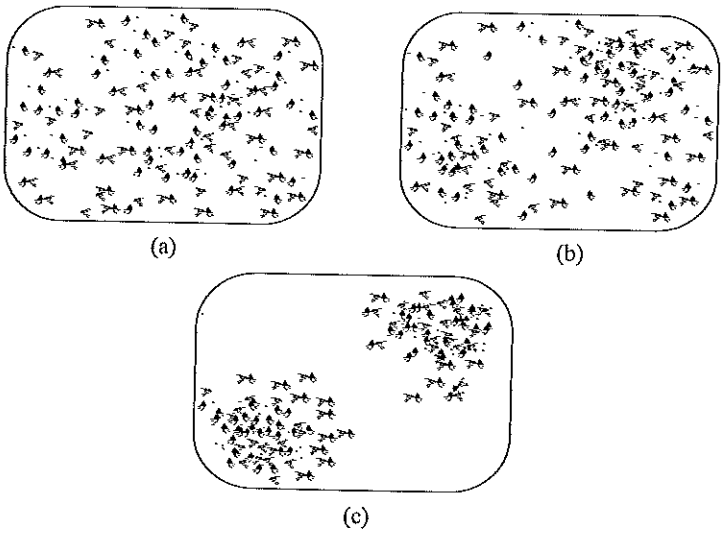

Figure 5.6: A qualitative example of cemetery formation in ants. (a) The corpses are initially randomly distributed over the arena. (b) After some time (e.g., a few hours) the workers start piling the corpses. (c) Clusters of corpses are formed by the worker ants.

The phenomenon observed in these experiments is the aggregation of dead bodies by workers. If dead bodies, or more precisely, items belonging to dead bodies, are randomly distributed in space at tbe beginning of the experiment, the workers will form clusters with these items within a few bours (Bonabeau, 1997). If the experimental arena is not sufficiently large or if it contains spatial heterogeneities, the clusters will be formed along the borders of the arena or along the heterogeneities. Figure 5.6 presents a qualitative illustration of how ants form clusters of corpses in a given arena. It is assumed that a number of corpses are spread over the arena and worker ants are allowed to pick them up and drop them somewhere within the arena.

The basic mechanism behind this type of clustering (aggregation) phenomenon is an attraction between dead items (corpses) mediated by ant workers. Small clusters of items grow by attracting more workers to deposit more dead items. It is this positive feedback mechanism that leads to the formation of larger and larger clusters of dead items. However, the exact individual behavior that leads to this positive feedback mechanism and the adaptive significance of corpse clustering are still unclear. Cleaning the nest by getting rid of dead items is an important task, though (Bonabeau, 1991; Bonabeau et al., 1999).

Another related phenomenon is larval sorting by the ant *Leptothorax unifasciatus*, a phenomenon widespread and broadly studied for this ant species (Franks and Sendova-Franks, 1992). Workers of this species gatber the larvae according to their size. All larvae tend to be clustered, with small larvae located in the center and larger larvae in the periphery of tbe brood.

Stigmergy

The stigmergic principle is also clear in this corpse-gathering behavior of ant colonies. Tbe observations show that ants tend to put the corpses of dead ants together in cemeteries in certain places far from the nest and that grow in size with time. If a large number of corpses are scattered outside the nest, the ants from the nest will pick them up, carry them about for a while and drop them. Within a short period of time, it can he observed that corpses are being placed in small clusters. As time goes by, the number of clusters decrease and the size of tbe remaining clusters increase, until eventually all the corpses are piled in one or two large clusters. Therefore, the increase in cluster size reinforces the enlargement of a cluster, and the reverse is also true.

5.2.5. Ant Clustering Algorithm (ACA)

In addition to path optimization to food sources, *data clustering* is another problem solved by real ants. Clustering problems bave already been explored in the previous chapter while describing the self-organizing networks and some of their applications. Before proceeding with the presentation of the clustering algorithm derived from models of cemetery organization and brood sorting in ant colonies, tbe reader should refer to Appendix B.4.5 for further comments on data clustering.

The Standard Ant Clustering Algorithm (ACA)

In order to model the two phenomena of dead body clustering and larval sorting in some ant species, Deneubourg et al. (1991) have proposed two closely related models. Both models rely on the same principles, but the first one (cemetery organization) is a special case of the second one (brood sorting). While the clustering model reproduces experimental observations more faithfully, the second one has given rise to more practical applications. In their study, a colony of ant-like agents, randomly moving in a bi-dimensional grid, was allowed to move basic objects so as to cluster them.

The general idea is that isolated items should be picked up and dropped at some other location where more items of that type are present. Assume that there is a single type of item in the environment, and a number of "artificial ants" whose role will be to carry an item and drop it somewhere in the environment. The probability p_p that a randomly moving ant, who is not currently carrying an item, picks up an item is given by

$$p_p = \left(\frac{k_1}{k_1 + f} \right)^2 \tag{5.11}$$

where f is the *perceived fraction of items* in the neighborhood of the ant, and k_1 is a threshold constant. For $f << k_1$, $p_p \approx 1$, thus the probability that the ant picks up an item is high when there are not many items in its neighborhood. In the case $f >> k_1$, $p_p \approx 0$, thus the probability that an item is removed from a dense cluster is small; that is, when there are many items in the neighborhood of an item it tends to remain there.

The probability p_d that a randomly moving ant, who is currently carrying an item, deposits that item in a certain position is given by

$$p_d = \left(\frac{f}{k_2 + f} \right)^2 \tag{5.12}$$

where k_2 is another threshold constant. In this case, if $f << k_2$, then $p_d \approx 0$, thus the probability that the ant deposits the item is low when there are not many items in its neighborhood. In the case $f >> k_2$, $p_d \approx 1$, thus the probability that the item is deposited in a dense cluster is high; that is, items tend to be deposited in regions of the arena where there are many items.

In order to use this theoretical model to generate an *ant clustering algorithm* (ACA), two main aspects have yet to be discussed. First, in which kind of environment are the ants going to move? That is, how to project the space of attributes so that it is possible to visualize clusters? Second, how is the function f (the perceived fraction of items) going to be defined?

Projecting the Space of Attributes into a Bi-Dimensional Grid

In the standard ant clustering algorithm (Lumer and Faieta, 1994), ants are assumed to move on a discrete 2-D board. This board may be considered as a bi-

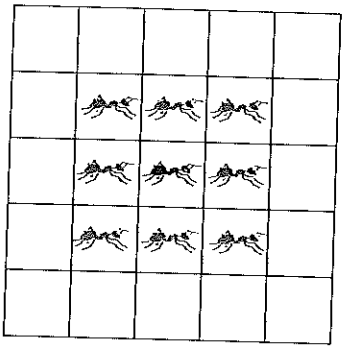

Figure 5.7: A bi-dimensional grid with 5 × 5 cells. An ant (dark ant in the center cell) is placed on a cell with its eight neighbors depicted (gray ants surrounding the center ant).

dimensional grid B of $m \times m$ cells. An ant is allowed to travel from one side of the board to another, meaning that the board is toroidal, and the ant perceives a surrounding region area, i.e., a squared neighborhood $\text{Neigh}_{(s \times s)}$ of $s \times s$ cells surrounding the current site (cell) r. In the example illustrated in Figure 5.7 the reference ant, the one in the middle, perceives a neighborhood of 3×3 cells.

This representation of the arena by a bi-dimensional board resembles the bi-dimensional grid of Kohonen's self-organizing map (Section 4.4.5), but is more akin to a bi-dimensional grid in a *cellular automaton* (Section 7.3). Therefore, each ant is modeled by an automaton able to move on the grid and displace objects according to probabilistic rules which are based only upon information of the local environment (neighborhood). The combined actions of a set (colony) of these automata (ants) will lead to the clustering in the same spatial region of the grid of data items belonging to the same classes.

In the algorithm proposed by Lumer and Faieta (1994), the bi-dimensional grid is used as a low dimensional space in which to project the space of attributes of the input data. In this case, the dimension is $z = 2$ so that, after the clustering by ants, clusters present the following property: intra-cluster distances should be small in relation to inter-cluster distances. Therefore, for $z = 2$, the input data are mapped onto a board composed of m^2 cells. Such a mapping, after the ant clustering process, must preserve the neighborhood inherent in the input data without creating too many new neighbors in z-dimensions that do not exist in the L-dimensional space corresponding to the original space of the input data set.

The idea of the ant clustering algorithm is thus to initially project the input data items in the bi-dimensional grid and scatter them randomly, as illustrated in Figure 5.8. The artificial ants (agents) are also spread all over the bi-dimensional grid, and allowed to move, pick up, and deposit items according to some similarity measure and given probabilities.

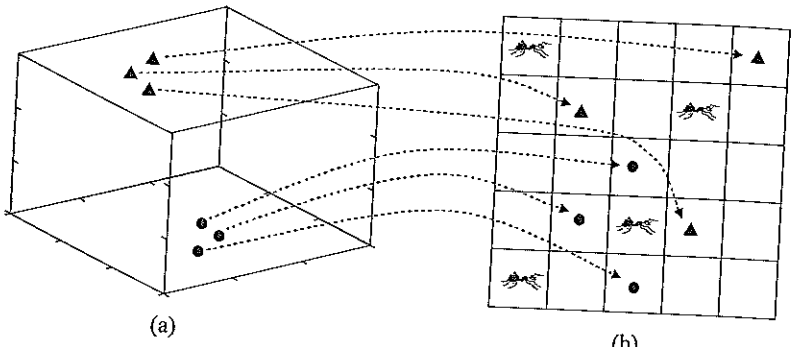

Figure 5.8: Projecting the input data into the bi-dimensional grid. (a) The input data lies in a three-dimensional space. (b) Before the ant clustering process starts, each input datum is projected into a random position of the bi-dimensional grid.

Defining the Perceived Fraction of Items: f

The second aspect that has to be accounted for is the definition of how f, the perceived fraction of items, is evaluated. Similarly to evolutionary algorithms (Chapter 3), f will be defined by the problem under study. For instance, in the context of robotic systems, Deneubourg et al. (1991) defined f as the quotient between the number M of items encountered during the last T time units and the largest possible number of items that can be encountered during these T time units.

Assuming that ants move in a bi-dimensional grid, the standard ACA can be summarized as in Algorithm 5.3. Let 'unloaded ant' be an ant not carrying an item, assume f is problem dependent, p_p and p_d are the probabilities of picking up and dropping off an item, respectively, and N is the number of ants. In this standard algorithm a cell can only be occupied by a single item.

Selected Applications from the Literature: A Brief Description

The standard ant clustering algorithm is described in Algorithm 5.3. Its applications in clustering range from exploratory data analysis to graph partitioning. To illustrate how to use the standard algorithm to different types of data, including to real-world data, it will be first described its application to simple numeric data and then to bioinformatics data. Both examples project the data into a bi-dimensional grid, but use different local density functions and picking up and dropping off probability functions. The examples focus on how to use the ACA standard algorithm to tackle these problems. Further details can be found in the cited literature.

Exploratory Data Analysis

Lumer and Faieta (1994) applied the standard ACA to exploratory data analysis. Their aim was to identify clusters in an unlabelled data set. To accomplish this

```
procedure [] = ACA(max_it, N, k1, k2)
    project every item i on a random cell of the grid
    place every ant on a random cell of the grid unoccupied
    by ants
    t ← 1
    while t < max_it do,
       for i = 1 to N do,  //for every ant
          compute f(x_i)
          if unloaded ant AND cell occupied by item x_i, then
             compute p_p(x_i)        //pick up probability
             pick up item x_i with probability p_p(x_i)
          else if ant carrying item x_i AND cell empty, then
             compute p_d(x_i)        //drop off probability
             deposit (drop) item x_i with probability p_d(x_i)
          end if
          move randomly to an unoccupied neighbor
       end for
       t ← t + 1
    end while
    print location of items
end procedure
```

Algorithm 5.3: Standard ant clustering algorithm (ACA).

task, the input data items belonging to an L-dimensional Euclidean space, $\mathbf{X} \in \Re^L$, were projected into a bi-dimensional grid Z^2, which can be considered as a discretization of a real space. The artificial ants were allowed to move about in this grid (discrete space) and perceive a surrounding region of area s^2 that corresponds to a square $\text{Neigh}_{(s\times s)}$ of $s \times s$ cells surrounding the current cell r, as illustrated in Figure 5.7.

The authors suggested the following function f to calculate the local density of items in the neighborhood of object $\mathbf{x}_i$ situated at site r:

$$f(\mathbf{x}_i) = \begin{cases} \dfrac{1}{s^2} \displaystyle\sum_{\mathbf{x}_j \in \text{Neigh}_{(s\times s)}(r)} \left[1 - \frac{d(\mathbf{x}_i, \mathbf{x}_j)}{\alpha}\right] & \text{if } f > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.13}$$

where $f(\mathbf{x}_i)$ is a measure of the average similarity of object $\mathbf{x}_i$ with another object $\mathbf{x}_j$ in the neighborhood of $\mathbf{x}_i$, α is a factor that defines the scale of dissimilarity, and $d(\mathbf{x}_i,\mathbf{x}_j)$ is the Euclidean distance between two data items in their original L-dimensional space. The parameter α determines when two items should or should not be located next to each other. For instance, if α is too large, there is not much discrimination between two items, leading to the formation of clusters composed of items that should not belong to the same cluster; and vice-versa.

Note: the distance $d(\cdot,\cdot)$ between items is calculated in their original space, not in the projected space.

Equation (5.13) takes into account the neighborhood of a cell ($1/s^2$), and the relative distance between items in this neighborhood. It proposes that the smaller the distance between the item an ant is carrying and the items in its neighborhood, the higher the local density of items perceived and, thus, the higher the probability of this item being dropped in that position (cell). Inversely, the more different the item being carried is from the items on a given neighborhood, the smaller the likelihood that this item is dropped in that vicinity.

The probabilities of picking up or depositing an item are given by the following equations:

$$p_p(\mathbf{x}_i) = \left(\frac{k_1}{k_1 + f(\mathbf{x}_i)}\right)^2 \tag{5.14}$$

$$p_d(\mathbf{x}_i) = \begin{cases} 2f(\mathbf{x}_i) & \text{if } f(\mathbf{x}_i) < k_2 \\ 1 & \text{otherwise} \end{cases} \tag{5.15}$$

where k_1 and k_2 are two constants playing similar roles to the constants in Equation (5.11) and Equation (5.12) respectively.

To illustrate the performance of this algorithm, the authors applied it, with p_p and p_d determined by Equation (5.14) and Equation (5.15), to an artificially generated set of points in $\Re^2$ (see Computational Exercise 3, Section 5.6.2).

Although the results obtained by Lumer and Faieta (1994) with the standard ant clustering algorithm when applied to the simple benchmark problem of Computational Exercise 3 were good, the authors observed the presence of more clusters in the projected space than in the original distribution of data. In order to alleviate this problem, they proposed a number of variations in their initially proposed algorithm: ants with different speeds, ants with memory, and behavioral switches:

1) *The use of ants with different moving speeds*: each ant in the colony is allowed to have its own speed. The speed of the swarm is distributed uniformly in the interval $[1, v_{max}]$, where $v_{max} = 6$ is the maximal speed of an ant, and v corresponds to the number of grid units (cells) walked per time unit by an ant along a given grid axis. This pace influences the probability an ant picks up or drops a given item according to the following equation:

$$f(\mathbf{x}_i) = \begin{cases} \dfrac{1}{s^2} \displaystyle\sum_{\mathbf{x}_j \in \text{Neigh}_{(s \times s)}(r)} \left[1 - \frac{d(\mathbf{x}_i, \mathbf{x}_j)}{\alpha + \alpha(v-1)/v_{max}}\right] & \text{if } f > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.16}$$

Therefore, fast moving ants are not as selective as slow ants in their estimation of the average similarity of an object to its neighbors.

2) *A short-term memory for the ants*: ants can remember the last m items they have dropped along with their locations. Each time an item is picked up, it is compared to the elements in the ant's memory, and the ant goes toward the location of the memorized element most similar to the one just collected. This behavior leads to the formation of a smaller number of sta-

tistically equivalent clusters, because similar items have a lower probability of starting a new cluster.

3) *Behavioral switches*: ants are endowed with the capability of destroying clusters that have not been reinforced by the piling up of more items during some specific time interval. This is necessary because items are less likely to be removed as clusters of similar objects are formed.

Bioinformatics Data

As discussed previously, ant clustering algorithms often generate a number of clusters that is much larger than the natural number of clusters in the data set. Besides, the standard ACA does not stabilize in a particular clustering solution; it constantly constructs and deconstructs clusters during adaptation. To overcome these difficulties, Vizine et al. (2005) proposed three modifications in the standard ACA: 1) a cooling schedule for k_1; 2) a progressive vision field that allows ants to 'see' over a wider area; and 3) the use of a pheromone function added to the grid as a way to promote reinforcement for the dropping of objects at denser regions of the grid.

Concerning the cooling schedule for k_1, the adopted scheme is simple: after one cycle has passed (10,000 ant steps), the value of the parameter k_1 starts being geometrically decreased, at each cycle, until it reaches a minimal allowed value, $k_{1\min}$, which corresponds to the stopping criterion for the algorithm:

$$\begin{aligned} k_1 &\leftarrow 0.98 \times k_1 \\ k_{1\min} &= 0.001 \end{aligned} \tag{5.17}$$

In the standard ACA, the value of the perceived fraction of items, $f(\mathbf{x}_i)$, depends on the vision field, s^2, of each ant. The definition of a fixed value for s^2 may sometimes cause inappropriate behaviors, because a fixed perceptual area does not allow the distinction of clusters with different sizes. To overcome this difficulty, a progressive vision scheme was proposed (Vizine et al., 2005). When an ant i perceives a 'large' cluster, it increments its perception field (s_i^2) up to a maximal size. A large cluster is detected using function $f(\mathbf{x}_i)$; when $f(\mathbf{x}_i)$ is greater than a given threshold θ, then the vision s^2 of an ant is incremented by n_s units until it reaches its maximum value:

$$\begin{aligned} \text{If } f(\mathbf{x}_i) > \theta \quad &\text{and} \quad s^2 \le s^2_{\max} \\ &\text{then } s^2 \leftarrow s^2 + n_s \end{aligned} \tag{5.18}$$

where $s^2_{\max} = 7 \times 7$ and $\theta = 0.6$ in their implementation.

Inspired by the way termites use pheromone to build their nests (Section 2.2.3), Vizine et al. (2005) proposed adding pheromone to the objects the ants carry and allowing the transference of this pheromone to the grid when an object is dropped.

The pheromone added to the carried items lead to the introduction of a pheromone level function $\phi(i)$ in the grid, which corresponds to how much pheromone is present at each grid cell i. During each iteration, the artificial pheromone $\phi(i)$

pheromone $\phi(i)$ at each cell of the grid evaporates at a fixed rate. This pheromone influences the picking and dropping probabilities as follows:

$$P_p(i) = \frac{1}{f(i)\phi(i)}\left(\frac{k_1}{k_1 + f(i)}\right)^2 \tag{5.19}$$

$$P_d(i) = f(i)\phi(i)\left(\frac{f(i)}{k_2 + f(i)}\right)^2 \tag{5.20}$$

Note that according to Equation (5.19) the probability that an ant picks up an item from the grid is inversely proportional to the amount of pheromone at that position and also to the density of objects around i. The reverse holds for Equation (5.20); the probability of an ant dropping an item on the grid is directly proportional to the amount of pheromone at that position and to the density of objects around i.

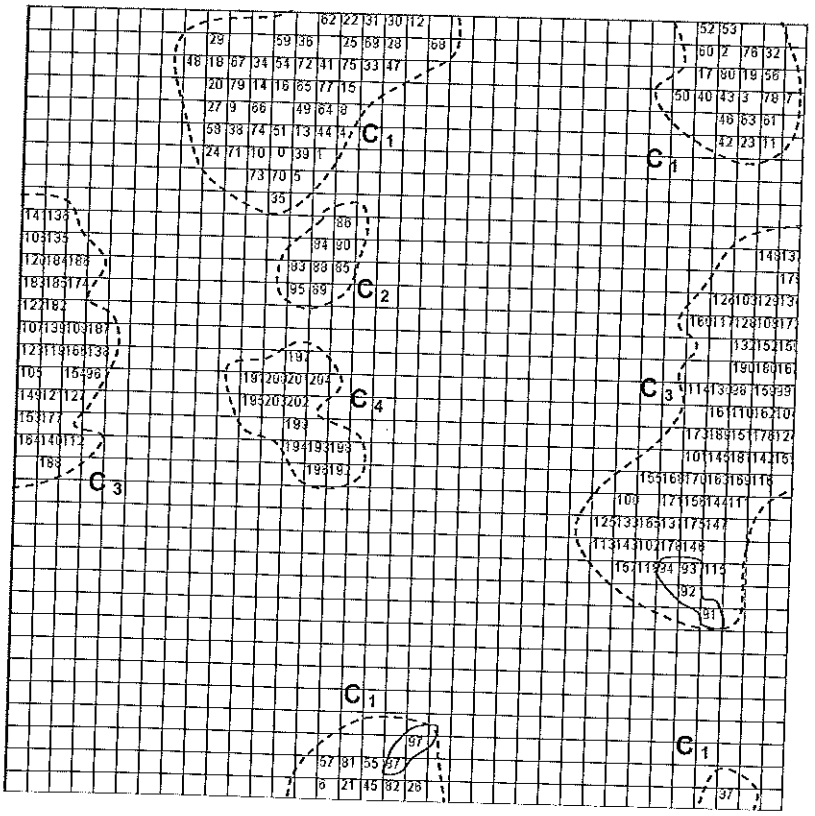

Figure 5.9: Final grid configuration for the bioinformatics data after the algorithm converged.

One of the problems the authors used to evaluate their proposal was in the field of bioinformatics, more specifically, the *yeast galactose* data set composed of 20 attributes (Yeung et al., 2003). Note that although each datum has dimension 20 all of them will be projected onto the bi-dimensional grid of the ACA. The authors used a subset of 205 data items, whose expression patterns or attributes reflect four functional categories (clusters) of genes formed by 83, 15, 93 and 14 genes (data). The authors used 10 ants to run the algorithm, $n_{ants} = 10$, a grid of size 35×35, $\alpha = 1.05$, $\theta = 0.6$, $k_1 = 0.20$, and $k_2 = 0.05$.

Figure 5.9 illustrates one grid configuration after the algorithm converged. Note the presence of four clusters of data (within dotted lines) and some misclassifications made by the algorithm (within solid lines). The numbers indicate the label of each datum in the data set. Note also the toroidal nature of the grid; that is, some clusters cross from one side of the picture to another, both from top to bottom and from left to right.

Scope of Ant Clustering Algorithms

The examples of application and the nature of the ant clustering algorithm suggest its potentiality for clustering problems. In more general terms, ACAs are suitable for exploratory data analysis, in which an unlabelled set of data is available and some information must be extracted from it. For instance, useful information is the degree of similarity between items and how to infer the cluster a new item (unknown to the system) belongs to.

It was discussed that the standard ACA fails in determining the appropriate number of clusters. However, some modifications can be introduced in the standard algorithm so as to make it suitable for determining a more accurate number of clusters in relation to the actual clusters of the input data set.

The ACA performs a dimensionality reduction in the space of attributes of the input data set into a bi-dimensional grid. This may be a useful approach when dealing with spaces of very high dimension, for it allows a visualization of neighborhood (similarity) relationships between the data items in the data set. Nevertheless, it may also cause some loss of information as a consequence of dimensionality reduction.

From Natural to Artificial Ants

Ant clustering algorithms were developed as a generalization of models of dead body clustering and brood sorting in ant colonies. The main features of ant clustering algorithms are a colony of ants that move within a bi-dimensional grid used to project the items (e.g., data and graph vertices) to be clustered. Ants are allowed to move items throughout this grid in a probabilistic manner, which is proportional to the degree of similarity between items. Table 5.2 presents one interpretation of the biological terminology to the computational ant clustering algorithm.

Table 5.2: Interpretation of the biological terminology into the computational ant clustering algorithm (ACA).

Biology (Ant Clustering)	Ant Clustering Algorithm
Environment (Arena)	Bi-dimensional grid in which items are projected and ants are allowed to move
Ant	Agent-like entity capable of moving about, picking up, and depositing items on the grid
Ant colony	Population (colony or swarm) of cooperating individuals known as artificial ants
Dead bodies or larvae	Items or objects (e.g., data patterns and vertices of a graph)
Pile (heap) of dead bodies	Clusters of items
Visibility of an ant	Perceived fraction of items f

5.2.6. Summary of Swarm Systems Based on Social Insects

This section discussed the behavior of social insects, focusing the case of ant colonies. It was observed that several different patterns of behavior can be seen in nature, from foraging for food to cemetery organization. These different behaviors lead to several computational tools and systems with various applications, such as combinatorial optimization and data analysis.

The majority of ant colony optimization algorithms are used to perform some sort of graph search associated with a combinatorial optimization problem. During the operation of an ACO algorithm, a colony of artificial ants is initialized in randomly chosen nodes of the graph G, and is allowed to move through adjacent nodes of this graph. The move of an ant from one node to another follows a probabilistic transition rule that is proportional to the pheromone level of a neighbor edge and the visibility (heuristic desirability) of the ant. By moving, ants incrementally build solutions to the problem and deposit pheromone on the edges traversed. This pheromone trail information will direct the search process of other ants, indicating that a certain route has been taken by (an)other ant(s). Pheromone trails are allowed to evaporate; that is, the level of pheromone of an edge is decreased over time, thus avoiding a rapid convergence towards suboptimal solutions.

The cemetery organization and larval sorting behavior of ants have led to the proposal of a theoretical model capable of reproducing a similar behavior. This model then led to the development of the ant clustering algorithm with potential applicability to exploratory data analysis, among other clustering tasks. The algorithm works by projecting the objects to be clustered into a bi-dimensional grid and probabilistically placing these objects onto the grid according to the density of similar items in a given region of the grid. The main aspects to be defined in this algorithm are the probability of placing or moving an item at or from a given position (site or cell) of the grid, and how to define the density function of items in this area of the grid.